Partial Case-Marking in Japanese Stripping/Sluicing: A Dynamic Syntax Account

Tohru Seraku
Hankuk University of Foreign Studies
81, Oedae-ro, Cheoin-gu, Yongin-si, Gyeonggi-do
449-791, Korea
seraku@hufs.ac.kr

Abstract

This article presents novel data on partial case-marking in Japanese stripping/sluicing: only the final NP in multiple stripping/sluicing may lack a case particle. These data challenge previous works that assign radically distinct structures to stripping/sluicing depending on whether or not case-marking is involved. These case-marking patterns are reducible to incremental growth of semantic representation, formalised in Dynamic Syntax: each NP is parsed at an 'unfixed' node, and this structural uncertainty must be resolved before another unfixed node is introduced.

1 Introduction

There is a growing body of research on ellipsis in Japanese (Hiraiwa & Ishihara 2012 and references therein). Stripping is a relatively understudied type of elliptical construction (Fukaya 2007, Fukaya & Hoji 2003, Fukui & Sakai 2003, Sakai 2000; see also Hankamer & Sag 1976). As shown in (1)B, stripping consists of the NP *Mary* and the copula *da*, where case-marking of *Mary* is optional.

(1) A: *Tom-ga ringo-o tabe-ta-yo*.
T-NOM apple-ACC eat-PAST-SFP
'Tom ate apples.'
B: *Iya*, *Mary*(*-ga*) *da*.
no M(-NOM) COP
'No, Mary.' (= 'No, Mary ate apples.')

Japanese also allows "multiple stripping." That is, the pre-copula part may involve more than one NP:

(2) A: *Tom-ga ringo-o tabe-ta-yo*.
T-NOM apple-ACC eat-PAST-SFP
'Tom ate apples.'
B: *Iya*, *Mary-ga nashi-o da*.
no M-NOM pear-ACC COP
'No, Mary, pears.' (= 'No, Mary ate pears.')

The most elaborated analysis of stripping is found in Fukaya (2007), the main claim being that case-marked and case-less stripping must be structurally distinguished. According to Fukaya, movement is relevant only to case-marked stripping.[1]

What has not been noted in previous studies is that when there are multiple NPs in stripping, only the **final** NP may be case-less (see Section 4 for details). Compare (2)B with (3)B, where the final NP *nashi* (= 'pear') may be case-less, but not the non-final NP *Mary*.[2]

(3) A: *Tom-ga ringo-o tabe-ta-yo*.
T-NOM apple-ACC eat-PAST-SFP
'Tom ate apples.'
B: *Iya*, *Mary**(*-ga*) *nashi da*.
no M(-NOM) pear COP
'No, Mary, pears.' (= 'No, Mary ate pears.')

[1] This non-uniform analysis is based on the observation that only case-marked stripping is sensitive to "islands" (Fukaya 2007). Seraku (2013) shows that our account captures the island-(in)sensitivity patterns of stripping by means of the 'LINK' mechanism (Cann et al. 2005).

[2] For some speakers, acceptability slightly drops with the string *Mary-ga nashi da*, but what is essential is that it is much more acceptable than the string *Mary nashi-o da* and the string *Mary nashi da*. The same type of remark also applies to the data in Sections 4 and 5.

28th Pacific Asia Conference on Language, Information and Computation pages 93–100

This partial case-marking phenomenon raises two problems for previous works. First, (3)B manifests case-marked and case-less stripping **at the same time**; that is, the single string contains the case-marked NP *Mary-ga* and the case-less NP *nashi*. It is thus not obvious how (3)B may be handled by the past **non-uniform** account that posits radically distinct structures depending on whether or not an NP in stripping is case-marked. Second, even if the first issue is sidestepped by stipulating a uniform syntactic structure for the two types of stripping, the question still remains of why **only** the final focus may lack a case particle.

The aim of this article is to show that the two recalcitrant puzzles are solved in a framework that directly reflects the incrementality of processing a string online, as modelled in Dynamic Syntax (DS) (Cann et al. 2005, Kempson et al. 2001, 2011).

Section 2 sets out the DS framework. Section 3 offers a unified analysis of stripping, and Section 4 deals with multiple stripping. Section 5 points out that the case-marking patterns of stripping are also found in sluicing, demonstrating that these sluicing data are amenable to our uniform analysis. Finally, Section 6 sums up the main results of this paper.

2 Dynamic Syntax (DS)

DS is a model of "competence," defined as a set of constraints on how to build an interpretation on the basis of incremental, word-by-word parsing online (Cann et al. 2005, Kempson et al 2001, 2011).[3] In the DS view of comprehension, the parser takes a string of words **left-to-right** and gradually builds an interpretation (represented as a **semantic** tree) **without** positing an independent level of syntactic structure. Syntax within DS is thus no more than a set of constraints on how to construct a semantic tree in real time.

DS semantic trees are binary-branching, where a right node is inhabited by a functor and a left node by an argument. Each node, if fully developed, is decorated with a semantic **content** and its semantic **type**. For instance, the parse of *Tom* decorates an argument node with the content *Tom′* and the type e, as in *Tom′* : e. Each node, if not fully developed, is decorated with **requirements**. The node to be decorated with *Tom′* : e is initially marked with ?e, which requires that the node will be decorated with the type e.

DS trees are progressively updated. The starting point is a root node with the requirement ?t, which requires that this node will be propositional. This initial state is defined as an **AXIOM** (see (5)). Once a root node is set out, it is subsequently updated by running **lexical** actions (triggered by the parse of a lexical item) or optionally running **general** actions.

An essential example of general actions is the introduction of an "unfixed" node, a node whose structural position is initially underspecified and will be resolved at a later point. Of note is LOCAL *ADJUNCTION, which introduces a **locally**-unfixed node decorated with the requirement **?e**.[4]

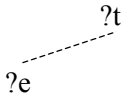
For an illustration, consider how a semantic tree is built incrementally by parsing (4) left-to-right.

(4) *Tom-ga* *hashi-tta*.
T-NOM run-PAST
'Tom ran.'

An initial state is the AXIOM (5), where ?t requires that this node will be decorated with a type-t (i.e. propositional) content. This is then updated to (6) by performing LOCAL *ADJUNCTION. This general action introduces an unfixed node; the positional uncertainty is expressed by a dashed line.

(5) AXIOM

(6) LOCAL *ADJUNCTION

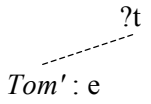


The unfixed node is decorated by the parse of *Tom*, triggering the actions to annotate the node with the content *Tom′* and the type e, as in (7). At this stage, the node is still unfixed, and it is the parse of the nominative case particle *ga* that fixes the structural underspecification, marking it as a subject node (i.e. the type-e node immediately dominated by the root node). The result of this resolution process is visually expressed in (8), where the dashed line has become a solid one.

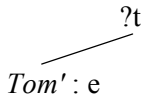
[3] DS also models language production with the same machinery as used for language comprehension (Howes 2012 and references therein).

[4] Seraku (2013) argues that a type-e unfixed node is induced by LOCAL *ADJUNCTION alone in Japanese.

(7) Parsing *Tom*

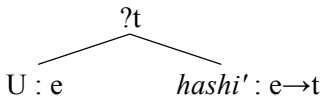


(8) Parsing *Tom-ga*



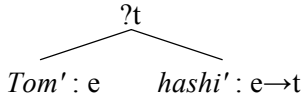
What comes next is *hashi* (= 'run'). Since Japanese is fully pro-drop, it is assumed that verbs project a propositional structure with argument slots. In the case of the intransitive verb *hashi*, it constructs a propositional structure where the subject argument is decorated with a place-holding meta-variable U.

(9) Output structure of parsing *hashi*



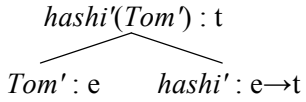
In (8), however, a subject node has already been created, and the argument slot provided by *hashi* collapses with this node. This is harmless since the argument slot is annotated with a meta-variable, a type of formula which is commensurate with any specified formula. Setting aside the tense suffix *ta* (see Cann 2011 and Seraku 2013 for a DS account of tense), the parse of *hashi* updates (8) into (10).

(10) Parsing *Tom-ga hashi*



Finally, functional application and type deduction take place. This process is modelled as the general action ELIMINATION. The tree (11) is a final state, representing the interpretation of the string (4).

(11) ELIMINATION



DS trees are "well-formed" iff no requirements are left in a tree, as in the tree (11). Furthermore, a string is "grammatical" iff there exists a sequence of tree updates from the AXIOM to a well-formed tree state (Cann et al. 2007).

3 A Uniform Account of Stripping

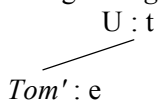
Building on Seraku's (2013) analysis of Japanese clefts, this section articulates a uniform account of case-marked and case-less stripping.

Firstly, we shall consider how the case-marked stripping (12)B (ignoring *iya* (= 'no')) is mapped onto a DS semantic tree incrementally.

(12) A: *Mary-ga hashi-tta-yo*.
M-NOM run-PAST-SFP
'Mary ran.'
B: *Iya*, *Tom-ga da*.
no T-NOM COP
'No, Tom.' (= 'No, Tom ran.')

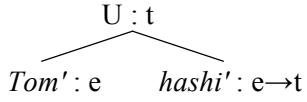
Starting with the AXIOM (5), the parse of (12)B up to *Tom-ga* leads to the tree (8). The next element in (12)B is the copula *da*. Seraku (2013) argues that *da* is a **type-t** pro-form, which posits a type-t meta-variable to be replaced with a propositional content.

(13) Parsing *Tom-ga da*



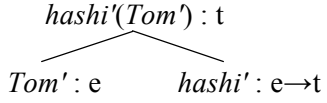
U is a type-t meta-variable. This tree state triggers the "re-use" of a previously-built type-t structure. Note that we have parsed the antecedent (12)A. In particular, when *hashi* (= 'run') was processed, a propositional structure with a subject slot was built. This is copied onto the present tree, updating (13) into (14), where the subject slot collapses with the node decorated with *Tom′* : e.

(14) Re-use of a previous structure



Finally, the parser runs ELIMINATION to clean up the tree, and the final state (15) correctly represents the interpretation of the stripping (12)B relative to the antecedent (12)A: 'No, Tom ran.'

(15) ELIMINATION

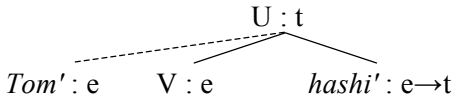


Let us turn to the case-less stripping (16)B. With the uniform nature of our account, a tree-update proceeds identically until *Tom* is parsed (see (7)).

(16) A: *Mary-ga hashi-tta-yo*.
M-NOM run-PAST-SFP
'Mary ran.'
B: *Iya*, *Tom da*.
no T COP
'No, Tom.' (= 'No, Tom ran.')

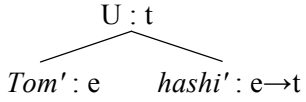
In (16), *Tom* is case-less, and thus the tree-update proceeds without resolving the unfixed node at this stage. The next expression is the copula *da*, which provides a type-t meta-variable, which triggers the "re-use" of the previous structure built by the parse of *hashi* in the antecedent.

(17) Re-use of a previous structure



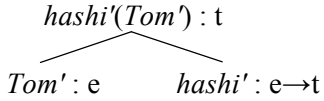
In (17), the node for *Tom* is unfixed. In general, an unfixed node may be merged with a fixed node of the same type. This structural merger is formulated as the general action UNIFICATION, which updates the tree (17) into (18).

(18) UNIFICATION



The unification process has fixed the node for *Tom* as a subject node. ELIMINATION outputs the final state (19), which is identical to (15), the tree for the case-marked stripping (12)B. This makes sure that the case-less stripping (16)B is truth-conditionally equivalent to the case-marked stripping (12)B.

(19) ELIMINATION



This section has developed a **uniform** account of case-marked and case-less stripping in the DS setting. The two types of stripping are mapped to **the same** tree, their difference being captured in terms of **how** a semantic tree is updated:

- In case-marked stripping, an unfixed node is fixed **lexically** by a case particle.
- In case-less stripping, it is fixed **non-lexically** by the general action UNIFICATION.

Let us close the present section by clarifying the notion of "focus." The NP in stripping is assumed to receive a focus (see Arregi 2010 and Merchant 2004). In DS, "focus" is not a primitive concept, but it emerges as an outcome of incremental tree growth (Cann et al. 2005). In stripping, the NP assigns a content value to an argument variable posited by a predicate in a presupposition clause. This saturation process evokes a focus effect as a result of incremental tree update (Seraku 2013).

4 Multiple Stripping

This section shows that our uniform treatment of stripping explains various types of data on multiple stripping data.

Within DS, each node is uniquely identified with respect to the other nodes in a tree (Blackburn & Meyer-Viol 1994). If **multiple** nodes are unfixed with respect to **the same** node, they will not be distinguishable. Thus, if supposedly distinct nodes are unfixed relative to the same node, they will lead to inconsistency in the node description.

(20) Unique-unfixed-node Constraint
If supposedly distinct nodes are unfixed with respect to the same node at a time, the node description becomes inconsistent.

This restriction is not a stipulation but a corollary of the tree logic (Blackburn & Meyer-Viol 1994). So, it plays a role in explaining linguistic puzzles cross-linguistically (Chatzikyriakidis & Kempson 2011, Gibson 2012).

Note that if two attempts to build a node with a different formula are possible only if the formulae are fully **commensurate**. In such a case, there will only be one such node. Consider UNIFICATION. In (18), the node decorated with the meta-variable V successfully merges with the node decorated with the formula *Tom*'. This is because a meta-variable is underspecified for its content and thus it is fully commensurate with any specified formula.

Based on the constraint (20), we shall address the case-marking issues of multiple stripping (see footnote 2). To being with, consider (21)B.

(21) A: *Mary-ga ringo-o tabe-ta-yo*.
M-NOM apple-ACC eat-PAST-SFP
'Mary ate apples.'
B: *Iya*, *Tom-ga nashi-o da*.
no T-NOM pear-ACC COP
'No, Tom, pears.' (= 'No, Tom ate pears.')

First, an unfixed node is introduced for *Tom*. This is immediately fixed by the case particle *ga*. At this point, an unfixed node is no longer in place, and an unfixed node may be once again introduced. This unfixed node is decorated by the second NP *nashi* (= 'pear') and resolved by the case particle *o*. So, the constraint (20) is not violated.

Next, consider the ungrammatical stripping data (22)B, where a case particle is dropped off *Tom* and *nashi* in (21)B.

(22) A: *Mary-ga ringo-o tabe-ta-yo*.
M-NOM apple-ACC eat-PAST-SFP
'Mary ate apples.'
B: **Iya*, *Tom nashi da*.
no T pear COP

In this example, an unfixed node for *Tom* cannot be resolved because (i) *Tom* is case-less and (ii) UNIFICATION cannot fire. Recall that UNIFICATION requires a **fixed type-e** node, but such a node is provided **after** the parse of the copula *da* triggers the re-use of a previous type-t structure. In short, UNIFICATION may be used for an unfixed node for the pre-copula NP alone. So, when an unfixed node is induced for the second NP *nashi*, there are two unfixed nodes relative to the same node at a time, violating the constraint (20).

Our analysis explains "partial case-marking," as illustrated in (23)B.

(23) A: *Mary-ga ringo-o tabe-ta-yo*.
M-NOM apple-ACC eat-PAST-SFP
'Mary ate apples.'
B: *Iya*, *Tom-ga nashi da*.
no T-NOM pear COP
'No, Tom, pears.' (= 'No, Tom ate pears.')

In this case, an unfixed node for *Tom* is resolved immediately by the nominative case particle *ga*, and an unfixed node can be safely introduced for the second NP *nashi*. This unfixed node cannot be resolved lexically since *nashi* lacks a case particle, but it can be resolved non-lexically by the general action UNIFICATION after the parse of *da*. So, there are no multiple unfixed nodes at a time, and the string is correctly predicted to be grammatical.

The analysis also predicts the ungrammaticality of (24)B, which exhibits the reversed case-marking pattern from (23)B.

(24) A: *Mary-ga ringo-o tabe-ta-yo*.
M-NOM apple-ACC eat-PAST-SFP
'Mary ate apples.'
B: **Iya*, *Tom nashi-o da*.
no T pear-ACC COP

These data are readily explained: an unfixed node for *Tom* cannot be fixed since (i) *Mary* is case-less and (ii) UNIFICATION cannot fire. Thus, the parser has to induce another unfixed node for the second NP *nashi*. This violates the constraint (20).

Our DS account is further corroborated by the multiple stripping with three NPs.

(25) A: *Tom-ga Mary-ni ringo-o age-ta-yo*.
T-NOM M-DAT apple-ACC give-PAST-SFP
'Tom gave apples to Mary.'
B: *Iya*, *Peter-ga Nancy-ni nashi-o da-yo*.
no P-NOM N-DAT pear-ACC COP-SFP
'No, Peter, to Nancy, pears.' (= 'No, Peter gave pears to Nancy.')
B': *Iya*, *Peter-ga Nancy-ni nashi da-yo*.
no P-NOM N-DAT pear COP-SFP

(25)B is grammatical since every unfixed node is immediately resolved by a particle. That is, there is only a single unfixed node at a time. (25)B' is also grammatical since an unfixed node for every non-final NP (i.e. *Peter*, *Nancy*) is immediately fixed by a particle, and an unfixed node for the final NP (i.e. *nashi*) is resolved by UNIFICATION after *da* is parsed. Once again, there is only a single unfixed node at a time. By contrast, the other case-marking patterns are ruled out: (i) only *Peter* is case-less, (ii) only *Nancy* is case-less, (iii) only *Peter* and *Nancy* are case-less, (iv) only *Peter* and *nashi* are case-less, (v) only *Nancy* and *nashi* are case-less, and (vi) every NP is case-less. In these cases, there are necessarily multiple unfixed nodes at a time.

Our uniform analysis explains the case-marking patterns of stripping as an outcome of incremental tree growth: an NP in stripping is processed at an unfixed node, and each unfixed node must be fixed before another unfixed node is introduced.

5 Extensions to Sluicing

There is a construction that is similar to stripping: sluicing (e.g. Hiraiwa & Ishihara 2012, Kizu 2005, Nishiyama et al. 1996, Takahashi 1996; see also Ross 1969). In this section, we note that the case-marking patterns of stripping are carried over into sluicing, and contend that our analysis of stripping is extended to various sluicing data.

In (26), the second clause exemplifies sluicing. As indicated in the parentheses, the case particle *ga* is optional, as in the case of stripping.

(26) *Paatii-de dareka-ga kyoku-o*
party-at someone-NOM song-ACC
uta-tta-ga, *boku-wa* [*dare*(*-ga*)*-ka*]
sing-PAST-but I-TOP [who(-NOM)-Q]
omoida-se-nai.
remember-can-NEG
'Someone sang a song at a party, but I cannot remember who sang a song.'

Multiple sluicing is also possible, as shown in (27). Of particular note is that in the sequence of *wh*-items, a case particle may be dropped off the final *wh*-item alone (in the present case, *nani*).

(27) *Paatii-de dareka-ga nanika-o*
party-at someone-NOM something-ACC
uta-tta-ga, *boku-wa* [*dare**(*-ga*)
sing-PAST-but I-TOP [who(-NOM)
nani(*-o*) *da-tta-ka*] *omoida-se-nai*.
what(-ACC) COP-PAST-Q] remember-can-NEG
'Someone sang something at a party, but I cannot remember who sang what.'

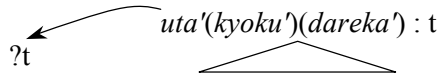
The tendency in the past literature is to assign a radically different structure to sluicing depending on whether a *wh*-phrase is case-marked (Fukaya 2007, 2013; see also Takahashi 1996). Such non-uniform analyses are challenged by (27), where a single sluicing involves a case-marked *wh*-phrase and a case-less *wh*-phrase simultaneously. Further, even if it is possible to invent a new mechanism which allows case-marked and case-less *wh*-items in a single clause, it remains the mystery why only the final *wh*-phrase may be case-less.

5.1 A Uniform Account of Sluicing

Our analysis of sluicing is essentially the same as that of stripping, but there are two new ingredients. First, the content of a *wh*-phrase is a "WH-meta-variable." Unlike usual meta-variables, WH-meta-variables do not have to be saturated (Kempson et al. 2001). Second, sluicing involves the embedding of clauses; within DS, this is analysed by inducing an unfixed node of **type-t**. Building on Cann et al. (2005), Seraku (2013) claims that such an unfixed node is induced by *ADJUNCTION in Japanese.

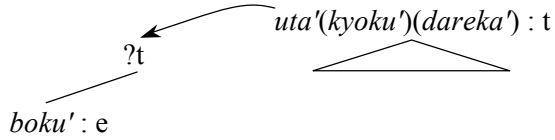
Let us first consider (26). The parse of the pre-*ga* clause results in a propositional structure. This is associated with another, emergent propositional structure by the parse of *ga* (= 'but'). Formally, this structure pairing is instantiated as a "LINK" relation, as visually expressed by a curved arrow. (The exact LINK mechanism is not relevant to our discussion; for details, see Cann et al. 2005 and Kempson et al. 2001). In (28), the adjunct *paatii-de* (= 'at a party') is neglected for brevity, and the internal structure is schematised as a triangle.

(28) Parsing *Dareka-ga kyoku-o uta-tta-ga*



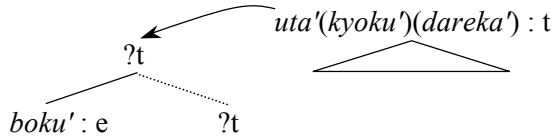
Then, the emergent propositional structure with ?t is fleshed out by the parse of the sluicing string. The parse of *boku-wa* leads to the usual structure-update: LOCAL *ADJUNCTION induces an unfixed node of type-e; this unfixed node is decorated by the matrix subject *boku* (= 'I'); finally, the node is resolved as a subject node by the topic marker *wa*.

(29) Parsing (26) up to *boku-wa*



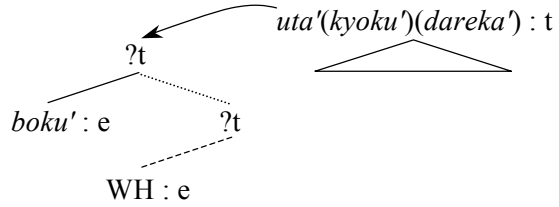
It is time to parse the *wh*-item *dare* (= 'who'). This is where the new ingredients come into place. First, *ADJUNCTION induces an unfixed node of type-t (expressed by a dotted line), allowing the parser to built an embedded propositional structure.

(30) *ADJUNCTION



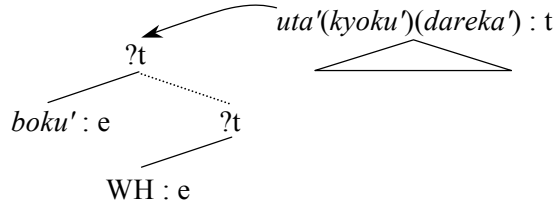
Second, LOCAL *ADJUNCTION fires to introduce an unfixed node of type-e. This node is decorated by the parse of the *wh*-phrase *dare*. As illustrated in (31), the content of *dare* is a WH-meta-variable. The unfixed node for *dare* may be resolved in two ways depending on the case-marking of *dare*.

(31) Parsing the string (26) up to *dare*



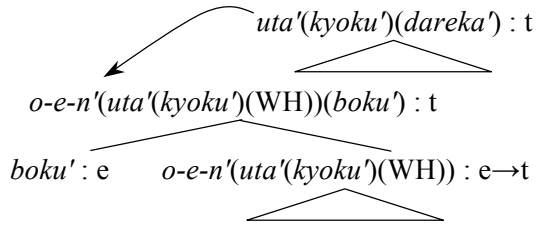
Case-marked sluicing: When *dare* is marked with the nominative case particle *ga*, the unfixed node for *dare* is immediately fixed as a subject node.

(32) Parsing the string (26) up to *dare-ga*



The next item *da* provides a type-t meta-variable, which triggers the re-use of the structure built by *uta* (= 'sing') in the first clause. With respect to this clause, the internal argument slot of *uta′* is saturated as *kyoku′*. As for the external argument slot, it collapses with the WH-meta-variable. Then, *omoidas-e-nai* (= 'cannot remember') fleshes out the higher ?t-decorated structure. This involves the creation of a type-t node as an internal argument. This type-t node is merged with the unfixed, lower type-t node by means of UNIFICATION. Finally, ELIMINATION is run, and the final state (33) holds, where *o-e-n′* is the content of *omoidas-e-nai.*

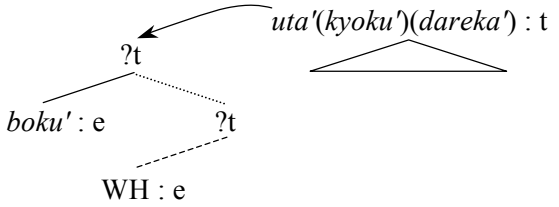
(33) ELIMINATION



Case-less sluicing: The tree state (33) holds even when the case particle *ga* is not attached to the *wh*-phrase *dare*. That is, irrespective of case-marking, uniformity in our analysis remains intact.

To begin with, the parse of (26) up to the *wh*-phrase *dare* yields (31), repeated as (34).

(34) Parsing the string (26) up to *dare*



Given that a case particle is absent, the tree-update proceeds without resolving the unfixed node for *dare*. The unfixed node gets resolved as a subject node by UNIFICATION after the copula *da* is parsed. This is because *da* triggers the re-use of a previous propositional structure, where there is a fixed node of type-e, with which the unfixed node of type-e is merged. The rest of the process is as usual, and the tree update ends with the final state (33). In this way, the identical final tree state holds no matter whether case-marking is encompassed in sluicing.

There is a remaining problem for our analysis of sluicing. Unlike stripping, the copula *da* in sluicing may be omitted (Nishiyama et al. 1996). Since *da* plays an important role in our account, it must be clarified why *da* may be dropped in sluicing but not stripping. This is a residual for future work.

5.2 Multiple Sluicing

The relevant data are repeated here as (35).

(35) *Paatii-de dareka-ga nanika-o*
party-at someone-NOM something-ACC
uta-tta-ga, *boku-wa* [*dare**(*-ga*)
sing-PAST-but I-TOP [who(-NOM)
nani(*-o*) *da-tta-ka*] *omoida-se-nai*.
what(-ACC) COP-PAST-Q] remember-can-NEG
'Someone sang something at a party, but I cannot remember who sang what.'

The case-marking patterns in (35) are explained in our account; the analysis is essentially the same as the one given in Section 4, and brief expositions would suffice. Firstly, multiple sluicing is possible as long as each *wh*-phrase has an appropriate case

particle. This is because an unfixed node for each *wh*-phrase can be immediately resolved by a case particle. Second, a case particle may be dropped only if it is attached to a final *wh*-phrase. This is because UNIFICATION (i.e. the non-lexical action to resolve an unfixed node) is applicable to the final *wh*-word: (i) UNIFICATION requires a propositional structure with a fixed type-e node, (ii) such a structure is provided by the copula *da*, and (iii) *da* is parsed only after all *wh*-phrases are processed.

In a nutshell, our dynamic account integrates the two types of sluicing and predicts the distribution of case particles in terms of incremental parsing.

6 Conclusion

Our analysis of stripping and sluicing is uniform in two senses: (i) stripping/sluicing are treated by the same machinery and (ii) for each construction, no distinct structures are postulated. Further, we have revealed the partial-case-marking patterns for these ellipsis constructions, and have shown that they are amenable to our unitary account.

Acknowledgments

I greatly benefitted from discussions with Ash Asudeh, Mary Dalrymple, Ruth Kempson, Jieun Kiaer, and Lutz Marten. I would like to thank the anonymous PACLIC reviewers for their valuable comments on the earlier version of this article.

References

Arregi, K. 2010. Ellipsis in Split Questions. Natural Language and Linguistic Theory 28: 539-592.

Blackburn, P., Meyer-Viol, W. 1994. Linguistics, Logic, and Finite Trees. Bulletin of Interest Group of Pure and Applied Logics 2: 2-39.

Cann, R. 2011. Towards an Account of the Auxiliary System in English. In Kempson, R., et al. (eds.) The Dynamics of Lexical Interfaces. CSLI, Stanford.

Cann, R., Kempson, R., Marten, L. 2005. The Dynamics of Language: An Introduction. Elsevier, Oxford.

Cann, R., Kempson, R., Purver, M. 2007. Context-dependent Well-formedness. Research on Language and Computation 5: 333-358.

Chatzikyriakidis, S., Kempson, R. 2011. Standard Modern and Pontic Greek Greek Person Restrictions. Journal of Greek Linguistics 11(2): 127-66.

Fukaya, T. 2007. Sluicing and Stripping in Japanese and some Implications. Ph.D. dissertation, University of Southern California.

Fukaya, T. 2013. Island Insensitivity in Japanese and some Implications. In Merchant, J., Simpson, A. (eds.) Sluicing: Cross-linguistic perspectives. Oxford University Press, Oxford.

Fukaya, T, Hoji, H. 2003. Stripping and Sluicing in Japanese and their Implications. Bird, S. et al. (eds). Proceedings of the 18th WCCFL. Cascadilla Press, MA, Somerville.

Fukui, N., Sakai, H. 2003. The Visibility Guideline for Functional Categories. Lingua 113: 321-375.

Gibson, H. 2012. Auxiliary placement in Rangi. Ph.D. thesis, SOAS (University of London).

Hankamer, J., Sag, I. 1976. Deep and Surface Anaphora. Linguistic Inquiry 7: 391-426.

Hiraiwa, K., Ishihara, S. 2012. Syntactic Metamorphosis. Syntax 15: 142-180.

Howes, C. 2012. Coordinating in Dialogue. Ph.D. thesis, Queen Mary, University of London.

Kempson, R., Gregoromichelaki, E., Howes, C. 2011. The Dynamics of Lexical Interfaces. CSLI, Stanford.

Kempson, R., Meyer-Viol, W., Gabbay, D. 2001. Dynamic Syntax. Blackwell, Oxford.

Kizu, M. 2005. Cleft Constructions in Japanese Syntax. Palgrave, New York.

Merchant, J. 2004. Fragments and Ellipsis. Linguistics and Philosophy 27: 661-738.

Nishiyama, K., Whitman, J., Yi, E.-Y. 1996. Syntactic Movement of Overt *Wh*-Phrases in Japanese and Korean. In Akatsuka, N., et al. (eds.) Japanese/Korean Linguistics 5. CSLI, Stanford.

Ross, J. R. 1969. "Guess Who?" In Binnick, R. I. et al. (eds.) Papers from the 5th Regional Meeting of Chicago Linguistic Society. University of Chicago Press, Chicago.

Sakai, H. 2000. Predicate Ellipsis and Nominalization in Japanese. Proceedings of 2000 Seoul International Conference on Language and Computation. Korea University, Seoul.

Seraku, T. 2013. Clefts, Relatives, and Language Dynamics. D.Phil. thesis, University of Oxford.

Takahashi, D. 1994. Sluicing in Japanese. Journal of East Asian Linguistics 3, 265-300.